Propositional Logic

0.1 Definitions

- Implication: When P is true, Q must be true. False implies true.
- $P \implies Q \equiv \neg P \vee Q$
- If σ is an assignment and φ is a formula, then $\sigma \models \varphi$ (sigma satisfies phi) means φ is true under assignment σ.
- A formula φ is satisfiable if there exists an assignment σ such that $\sigma \models \varphi$. Otherwise, we say it's unsatisfiable.
- A formula is valid (aka a tautology) if for every assignment σ, $\sigma \models \varphi$.
- Two formulas φ and ψ are logically equivalent, denoted $\varphi \equiv \psi$, if for every assignment σ, $\sigma \models \varphi$ iff $\sigma \models \psi$.
- Formulas φ and ψ are equisatisfiable if φ is satisfiable iff ψ is satisfiable.

0.2 The Boolean Satisfiability Problem (SAT)

- Given a propositional formula φ, is φ satisfiable?
- Example: System M with input x and output y and an input-output specification $\varphi(x, y)$. If we can encode the computation of M as a formula $\psi_m(x, y)$ such that this formula holds iff y is the output of M on input x (ψ_m is called the input-output relation or transition relation of M), then we can decide whether M satisfies φ for all inputs using this formula:

  $$\chi = \psi_m(x, y) \wedge \neg\varphi(x, y)$$

  The formula is satisfiable iff $M \not\models \varphi$, ie there is some input x such that M violates φ on that input. So to verify M, we can check if χ is unsatisfiable.
- Usually when we refer to SAT, we are referring to the CNF-SAT problem, where the input formula is in conjunctive normal form (CNF).
- A formula is in CNF if it is a conjunction of clauses, where each clause is a disjunction (OR) of literals, and a literal is either a variable or the negation of a variable. Example:

  $$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_4) \wedge (x_2 \vee x_3 \vee \neg x_4)$$

- The top level operator must be an AND.
- Any formula can be converted to CNF.
- A useful transformation is the Tseitin transformation, which converts any formula φ to an equisatisfiable CNF formula ψ in linear time. The idea is to introduce a new variable for each subformula of φ and add clauses that enforce the equivalence between the new variable and the subformula.

Example:

$$\varphi = \neg(x \vee y) \vee (\neg z \wedge x)$$

List the cases for the clause (gate) (call the output the clause u):

$$x \implies \neg u$$

$$y \implies \neg u$$

$$\neg x \wedge \neg y \implies u$$

Using demorgans law, we can rewrite these as:

$$\neg x \vee \neg u$$

$$\neg y \vee \neg u$$

$$x \vee y \vee u$$

AND them all together to represent the first gate!

$$(\neg x \vee \neg u) \wedge (\neg y \vee \neg u) \wedge (x \vee y \vee u)$$

These three clauses uniquely determine u given values for x and y (among satisfying assignments). Repeating this for every gate in the circuit gives a set of clauses encoding how the whole circuit works. Then add one more clause asserting that the output wire is true. Now any satisfying assignment of the original formula can be extended to one satisfying the new formula, and conversely any satisfying assignment of the new formula satisfies the original.

- Since there is 1 gate per boolean operator, and a constant number of clauses per gate, the new CNF formula has size linear in the size of the original formula.
- Disjunctive Normal Form (DNF) is the dual of CNF, where the top level operator is OR and each clause is an AND of literals. Example:

$$(x_1 \wedge \neg x_2 \wedge x_3) \vee (\neg x_1 \wedge x_4) \vee (x_2 \wedge x_3 \wedge \neg x_4)$$